IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

On the Computation of Sensitivity Tubes

Andrea Pupa[1], Tommaso Belvedere[2], Cristian Secchi[1] and Paolo Robuffo Giordano[2]

***Abstract*—Achieving robust robot control requires explicit treatment of model uncertainties. Closed-loop sensitivity has emerged as a powerful tool to analyze how parameter errors map into state and input deviations through so-called "sensitivity tubes", traditionally built from *ellipsoidal* uncertainty sets and used to robustify system constraints. These ellipsoids, however, are themselves smooth approximations of underlying *hyperboxes* in the parameter space, leading to an inaccurate estimation of the parameter set. This paper extends that framework by proposing two new formulations that more precisely represent the real closed-loop behavior of the system through improved computation of the sensitivity tubes. The first constructs tubes directly from *hyperboxes*, exactly preserving the original parameter bounds but producing a non-differentiable description. The second employs *superquadrics*, which smoothly approximate the hyperbox with user-tunable fidelity while preserving differentiability, as in the ellipsoidal case. Both methods are validated through an extensive simulation campaign, where the resulting input tubes ensure actuator constraints are respected. The results demonstrate that the new tubes better enclose the perturbed trajectories with respect to ellipsoidal ones, enhancing robustness for both online and offline trajectory planning.**

***Index Terms*—Planning under Uncertainty, Optimization and Optimal Control, Integrated Planning and Control**

I. INTRODUCTION

ROBOTS are now more widespread than ever across various fields. In industry, humans and robots collaborate in close proximity without the need for physical barriers. In operating rooms, robotic systems enhance surgical precision, particularly in procedures such as laparoscopic surgery. In agriculture, swarms of drones are deployed to monitor vast fields of crops efficiently. As a result, it is crucial to ensure that robots can perform the required tasks safely and reliably despite the unavoidable uncertainties affecting their models.

Various paradigms have been explored for controlling robots affected by model uncertainty. One approach involves offline identification of model parameters, but this often requires collecting large amounts of data, which can itself be unsafe when uncertainties are substantial [1]. Alternatively, online adaptive methods [2] can compensate for uncertainties or disturbances; however, they may degrade transient performance, compromise system stability, and prove unsuitable for tasks where parameters change abruptly. Robust control techniques [3], such as sliding mode control [4] or passivity-based methods [5], [6], offer another solution. However, these methods rely on specific model properties, often require high actuation bandwidth, and typically result in performance degradation even when parameters are well estimated (e.g., they are tuned for the worst case).

A complementary approach, which can be integrated with the aforementioned methods, involves planning and executing motions that result in inherently reduced sensitivity to uncertainties. In this context, closed-loop sensitivity has recently emerged as a valuable tool for handling uncertainties in system parameters [7]. This technique focuses on analyzing how variations in model parameters affect the behavior of a system when operating under feedback control, and can be used to generate intrinsically robust motion plans. The main advantage of this approach is that it does not require the design or use of a specific controller, allowing it to be successfully applied to many different cases. For instance, it has been effectively employed and validated experimentally in quadrotor trajectory tracking [8], energy tank initialization for manipulators [9], and sampling-based motion planning [10]. A significant development in this area is the introduction of *sensitivity tubes*, as proposed in [11]. These tubes represent an envelope that bounds the perturbed trajectories of the system states and inputs under the effect of parametric uncertainties. This has been demonstrated to be a powerful, direct approach to account for such uncertainties in safety critical offline planning [11] and robust Model Predictive Control [12]. To construct these tubes, one assumes that the model parameters lie within a bounded range centered around a nominal value—typically modeled as a hyperbox. The corresponding sensitivity matrices are then used to estimate how deviations from the nominal parameters influence the state and input trajectories. This approach effectively captures the worst-case impact of uncertainty and allows for robust constraint satisfaction in trajectory optimization. In [11], however, the parameter set is approximated by the ellipsoid inscribed in the hyperbox for obtaining a tractable and differentiable formulation of the tubes suitable for gradient-based trajectory optimization. Unfortunately, this ellipsoidal approximation can be quite restrictive since, as the number of parameters increases, the ellipsoid-to-hyperbox volume ratio tends to zero, excluding many valid parameter combinations near the corners. This under-approximates system perturbations and degrades predictions of uncertainty effects.

To address this issue, this work extends [11] by introducing

Manuscript received: March, 4, 2025; Revised May, 28, 2025; Accepted June, 27, 2025.

This paper was recommended for publication by Editor L. Pallottino upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the project ANR-20-CE33-0003 "CAMP" and by the National Recovery and Resilience Plan (NRRP), Mission 04 Component 2 Investment 1.5 – NextGenerationEU, Call for tender n. 3277 dated 30/12/2021. Award Number: 0001052 dated 23/06/2022.

[1] A. Pupa and C. Secchi are with the Department of Science and Methods of Engineering, University of Modena and Reggio Emilia, Italy. E-mail: {andrea.pupa, cristian.secchi}@unimore.it.

[2]T. Belvedere and P. Robuffo Giordano are with CNRS, Univ Rennes, Inria, IRISA, Campus de Beaulieu, 35042 Rennes Cedex, France. E-mail: {tommaso.belvedere, prg}@irisa.fr.

Digital Object Identifier (DOI): see top of this page.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

three methods for computing sensitivity tubes. The first computes tubes directly assuming parameter deviations lie within a hyperbox, avoiding set approximation but resulting in non-differentiable tubes. The second method is the one described in [11], where the hyperbox is approximated by an ellipsoid, but with an extended formal proof of the mapping from the parameter space to the state space. Lastly, the third method uses superquadrics to approximate the hyperbox with arbitrary precision [13], [14] while maintaining differentiability, offering a compromise between the previous two methods.

Summarizing, the main contributions of this work are:

- Two new methods to compute the sensitivity tubes: one based on hyperboxes, suitable for gradient-free optimization algorithms (such as MPPI), and one based on superquadrics, suitable for gradient-based optimization.
- The formal proofs of these two new methods as well as the one previously proposed in [11];
- A comprehensive validation of the new methods in a simulated environment, including also a comparison with [11], demonstrating better performance in encapsulating the real behavior of the system.

The paper is organized as follows: Sec. II provides the theoretical background on closed-loop sensitivity. The first part of Sec. III formally defines the sensitivity tube radius and presents the three methods for computing sensitivity tubes; while Sec. III-A–III-C detail the corresponding proofs for each method and may be skipped on an initial reading. Lastly, Sec. IV presents the experimental validation across different simulation environments, and Sec. V discusses conclusions and future work.

II. CLOSED-LOOP SENSITIVITY

For the reader's convenience, we now recall the main notion related to the closed-loop sensitivity and associated quantities. Consider a generic robot modeled by the following ordinary differential equation:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{p}), \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^{n_{\boldsymbol{x}}}$ represents the state and $\boldsymbol{u} \in \mathbb{R}^{n_{\boldsymbol{u}}}$ is the control input. Vector $\boldsymbol{p} \in \mathbb{R}^{n_{\boldsymbol{p}}}$, instead, represents a set of model or environmental parameters whose true values are not perfectly known. For example, in a robot arm, such parameters may represent the position of the center of mass of a link, the friction coefficient of the motors, or the elastic properties of the material it is interacting with.

In the context of trajectory tracking, consider an output function $\boldsymbol{y}(\boldsymbol{x}) \in \mathbb{R}^{n_{\boldsymbol{y}}}$, e.g., the manipulator pose, that needs to follow a desired reference trajectory $\boldsymbol{y}_d(t) \in \mathbb{R}^{n_{\boldsymbol{y}}}$, and an associated controller with generic expression

$$\begin{cases} \dot{\boldsymbol{\xi}} = \boldsymbol{g}(\boldsymbol{\xi}, \boldsymbol{x}, \boldsymbol{y}_d(t), \boldsymbol{p}_n, \boldsymbol{k}_c, t), \\ \boldsymbol{u} = \boldsymbol{h}(\boldsymbol{\xi}, \boldsymbol{x}, \boldsymbol{y}_d(t), \boldsymbol{p}_n, \boldsymbol{k}_c, t), \end{cases} \tag{2}$$

where $\boldsymbol{\xi} \in \mathbb{R}^{n_{\boldsymbol{\xi}}}$ are the internal states of the controller, $\boldsymbol{p}_n \in \mathbb{R}^{n_{\boldsymbol{p}}}$ are a *nominal* value for the model parameters, i.e., the estimated nominal value of $\boldsymbol{p}$ used by the controller that may differ from the real $\boldsymbol{p}$ due to the uncertainties, and $\boldsymbol{k}_c \in \mathbb{R}^{n_{\boldsymbol{k}}}$ are the controller gains. When $\boldsymbol{p} \neq \boldsymbol{p}_n$, the so-called *perturbed* case, the controller cannot perfectly realize the tracking task, e.g. the error may only converge to zero after some transient, or converge to a bounded set around the desired trajectory, and safety constraints might be violated even if the nominal trajectory is feasible. It is therefore crucial to develop a framework that, being aware of model uncertainties, can not only minimize their effects on the task execution but also (and perhaps most importantly) guarantee constraint satisfaction in the real (perturbed) case.

In order to address this problem, it is necessary to characterize how perturbations in the parameters $\boldsymbol{p}$ affect the system state $\boldsymbol{x}$, the input $\boldsymbol{u}$, and the output $\boldsymbol{y}$ trajectories in closed-loop during motion. To this aim, it is possible to exploit the *closed-loop sensitivity* notion introduced in [7] to describe how much the (closed-loop) system is locally sensitive to parametric uncertainties. Given the closed-loop system (1)–(2), one defines the so-called *state sensitivity matrix*:

$$\boldsymbol{\Pi}(t) = \frac{\partial \boldsymbol{x}(t)}{\partial \boldsymbol{p}}\Big|_{\boldsymbol{p}=\boldsymbol{p}_n} \in \mathbb{R}^{n_{\boldsymbol{x}} \times n_{\boldsymbol{p}}}, \tag{3}$$

and *input sensitivity matrix*:

$$\boldsymbol{\Theta}(t) = \frac{\partial \boldsymbol{u}(t)}{\partial \boldsymbol{p}}\Big|_{\boldsymbol{p}=\boldsymbol{p}_n} \in \mathbb{R}^{n_{\boldsymbol{u}} \times n_{\boldsymbol{p}}}, \tag{4}$$

which quantify how variations in the parameters $\boldsymbol{p}$ (w.r.t. the nominal $\boldsymbol{p}_n$) affect the states and inputs during motion. In the general case, a closed-form expression for $\boldsymbol{\Pi}(t)$ and $\boldsymbol{\Theta}(t)$ is not available. However, these quantities can be easily obtained by forward integrating the following system of differential equations:

$$\begin{cases} \dot{\boldsymbol{\Pi}}(t) = \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\boldsymbol{\Pi} + \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\boldsymbol{\Theta} + \dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{p}}, & \boldsymbol{\Pi}(t_0) = \boldsymbol{0}, \\ \dot{\boldsymbol{\Pi}}_{\boldsymbol{\xi}}(t) = \dfrac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}\boldsymbol{\Pi} + \dfrac{\partial \boldsymbol{g}}{\partial \boldsymbol{\xi}}\boldsymbol{\Pi}_{\boldsymbol{\xi}}, & \boldsymbol{\Pi}_{\boldsymbol{\xi}}(t_0) = \boldsymbol{0}, \\ \boldsymbol{\Theta}(t) = \dfrac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\boldsymbol{\Pi} + \dfrac{\partial \boldsymbol{h}}{\partial \boldsymbol{\xi}}\boldsymbol{\Pi}_{\boldsymbol{\xi}}, \end{cases} \tag{5}$$

where $\boldsymbol{\Pi}_{\boldsymbol{\xi}} \in \mathbb{R}^{n_{\boldsymbol{\xi}} \times n_{\boldsymbol{p}}}$ is the sensitivity of the controller states.

III. SENSITIVITY TUBES COMPUTATION

To compute sensitivity tubes, we assume that the actual parameters lie within a compact and known set $\boldsymbol{\mathcal{P}}$. The aim is to assess how variations in these parameters w.r.t. a nominal value affect the closed-loop trajectory. Such an analysis enables the construction of the so-called *sensitivity tubes*, a set of envelopes that bundles all possible perturbed trajectories resulting from parameter deviations, which is essential information for robust trajectory planning.

Rather than describing state/input tubes based on the geometric shape of $\boldsymbol{\mathcal{P}}$[1], the focus of the proposed analysis is in obtaining a general measure: the maximum deviation α that the *perturbed system* can exhibit along a given direction $\boldsymbol{n} \in \mathbb{R}^{n_{\boldsymbol{x}}}$. This deviation α represents the *tube radius* along that direction. It is worth underlining that, although the computed value of α will depend on the geometric shape of $\boldsymbol{\mathcal{P}}$, e.g. box versus ellipsoids, the definition of α as the maximum deviation along $\boldsymbol{n}$ is independent from the geometric shape $\boldsymbol{\mathcal{P}}$.

[1]Which is, for example, the case in [15], where the authors characterize tubes using ellipsoids, an approach that cannot be directly generalized to other geometric shapes.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

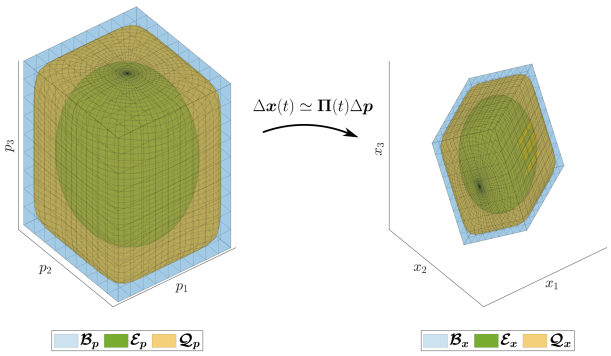

Fig. 1: Representation of the mapping obtained with the sensitivity matrix $\boldsymbol{\Pi}$. On the left the hyperbox $\boldsymbol{\mathcal{B}_p}$, the ellipsoid $\boldsymbol{\mathcal{E}_p}$, and the superquadric $\boldsymbol{\mathcal{Q}_p}$ in the parameter space. On the right, the corresponding shapes $\boldsymbol{\mathcal{B}_x}$, $\boldsymbol{\mathcal{E}_x}$, and $\boldsymbol{\mathcal{Q}_x}$ in the state space.

To carry out the following analysis, the set $\boldsymbol{\mathcal{P}}$ is assumed to be a symmetric hyperbox[2]. That is, each p_i can deviate from its nominal value $p_{n,i}$ by at most $\Delta p_{\max,i} > 0$, such that $p_i \in [p_{n,i} - \Delta p_{\max,i}, p_{n,i} + \Delta p_{\max,i}]$. This is a reasonable assumption as, in many applications, physical parameters can be associated with an uncertainty range due to measurement errors or task specifications, e.g., a minimum/maximum payload weight. Consequently, the set of the parameters can be defined as:

$$\boldsymbol{\mathcal{P}} := \boldsymbol{\mathcal{B}_p} = \{\boldsymbol{p} \in \mathbb{R}^{n_p} : -\Delta \boldsymbol{p}_{\max} \leq \boldsymbol{p} - \boldsymbol{p}_n \leq \Delta \boldsymbol{p}_{\max}\}, \quad (6)$$

with inequalities intended element-wise. In the following, three different methods to compute α are proposed and analyzed. Such methods exploit the closed-loop state sensitivity matrix $\boldsymbol{\Pi}$ to map finite perturbations from parameter space to state space. This mapping is graphically represented in Fig. 1. We note that, although the following derivations focus on the state space, the proposed method can be seamlessly applied to any quantity whose sensitivity matrix is available, e.g., the input space by using $\boldsymbol{\Theta}$, or any submanifolds arising from lower-dimensional functions of the state and/or inputs.

Let $\Delta \boldsymbol{x}(t) = \boldsymbol{x}(t) - \boldsymbol{x}_n(t)$: for small enough $\Delta \boldsymbol{p}$, $\Delta \boldsymbol{x}(t)$ can be linearly approximated as

$$\Delta \boldsymbol{x}(t) \simeq \boldsymbol{\Pi}(t)\Delta \boldsymbol{p}. \quad (7)$$

Starting from this linearization, the tube radius along an arbitrary direction can be computed for all the geometric shapes considered, namely the hyperbox, the ellipsoids, and the superquadric as:

$$\alpha_{\boldsymbol{\mathcal{B}}} = \|\boldsymbol{n}^\top \boldsymbol{\Pi} \boldsymbol{W}^{\frac{1}{2}}\|_1 \quad (8)$$

$$\alpha_{\boldsymbol{\mathcal{E}}} = \sqrt{\boldsymbol{n}^\top \boldsymbol{\Pi} \boldsymbol{W} \boldsymbol{\Pi}^\top \boldsymbol{n}} \quad (9)$$

$$\alpha_{\boldsymbol{\mathcal{Q}}} = \sqrt[\frac{2k}{2k-1}]{\boldsymbol{n}^\top \boldsymbol{\Pi} \boldsymbol{W}^{\frac{1}{2}} \left(\boldsymbol{W}^{\frac{1}{2}} \boldsymbol{\Pi}^\top \boldsymbol{n}\right)^{\circ \frac{1}{2k-1}}} \quad (10)$$

where $k \geq 1$, superscript $\circ$ denotes the element-wise power [16], e.g. $\Delta \boldsymbol{p}^{\circ j^\top} = \left[\Delta p_1^j, \ldots, \Delta p_{n_p}^j\right]$, and $\boldsymbol{W} = \operatorname{diag}\{\Delta p_{\max,i}^2\}$. The detailed proofs of (8)–(10) are presented in the following Sec. III-A–Sec. III-C.

The first method derives directly from the definition of $\Delta \boldsymbol{p}$.

[2]While symmetry is not strictly required, it simplifies the mathematical formulation.

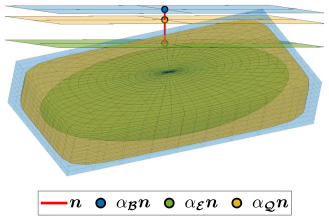

Fig. 2: Graphical representation of the tube radius α along a direction $\boldsymbol{n}$. Regardless of the analyzed geometric shape, α denotes the distance from the center of the shape to the tangent plane whose normal is $\boldsymbol{n}$.

Since equation (6) represents a hyperbox centered in $\boldsymbol{p}_n$, it can be demonstrated (see Section III-A), that the maximum state deviation occurs when the parameter deviations align with one of the hyperbox corners. However, mapping such a corner results in a non-differentiable expression for α, which is undesirable for gradient-based trajectory optimization algorithms.

To ensure differentiability, the original hyperbox can be approximated by its inscribed ellipsoid as proposed in [11]. Using matrix $\boldsymbol{\Pi}$, this ellipsoid in parameter space is mapped into a corresponding ellipsoid in state space. The tube radius is then obtained by evaluating the maximum variation of the ellipsoid along a generic direction $\boldsymbol{n}$.

However, the original set of parameter variations defined in (6) is an hyperbox: as the number of parameters increases, the portion of volume of the box occupied by the inscribed ellipsoid decreases which in turn implies an increased probability that the real parameter deviation will lie outside the ellipsoid. This results in tubes (and tube radii) that can underestimate the effects of uncertainties and lead to incorrect predictions in trajectory planning. Of course, one could approximate the hyperbox with the circumscribed ellipsoid. However, this solution would clearly lead to conservative behaviors.

The third method addresses this issue by utilizing superquadrics [13], specifically superellipsoids, a shape that more closely resembles the original hyperbox. As defined in Sect. III-C, the superellipsoid reduces to a standard ellipsoid when of order $k = 1$, while it converges to the hyperbox as $k \to \infty$.

It is worth emphasizing that, regardless of the analyzed geometric shape, the tube radius α always maintains the same geometric meaning. As illustrated in Fig. 2, from a geometric perspective, α represents the distance from the center of the analyzed shape, i.e., $\boldsymbol{p}_n$, to the tangent plane with normal vector $\boldsymbol{n}$.

Remark 1. For all three methods, the tube radius expression has been derived for a generic direction $\boldsymbol{n}$. Often, one needs to evaluate how each component of the state deviates from its nominal value. To evaluate this for the i-th component, one can set $\boldsymbol{n} = \boldsymbol{e}_i$, where $\boldsymbol{e}_i$ denotes the i-th vector of the canonical basis $(\boldsymbol{e}_1, \ldots, \boldsymbol{e}_{n_x})$ of $\mathbb{R}^{n_x}$. This enables, for instance, a direct check of whether the deviation of the i-th state component remains within its specified bounds in the worst-case scenario, thereby allowing robust constraint enforcement.

The detailed proofs of (8)–(10) for the three considered cases are now provided. Readers not interested in the technical

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

details may skip these sections and proceed directly to the results in Sec. IV.

A. Hyperbox

Assuming that the uncertain parameters belong to the set (6), the sensitivity tube radius can be computed by finding which vertex of the hyperbox generates the maximum state deviation along a given direction $\boldsymbol{n}$.

Proposition 1. *The maximum state deviation occurs when the real parameters lie at one of the vertices, depending on the closed-loop sensitivity matrix $\boldsymbol{\Pi}$, and the tube radius $\alpha_{\mathcal{B}}$ along a generic direction $\boldsymbol{n}$ can be computed using equation* (8)*.*

Proof. The goal is to obtain the maximum deviation $\alpha_{\mathcal{B}}$ such that:

$$\alpha_{\mathcal{B}} = \boldsymbol{n}^\top \Delta \boldsymbol{x} \ \wedge \ -\Delta \boldsymbol{p}_{\max} \leq \Delta \boldsymbol{p} \leq \Delta \boldsymbol{p}_{\max}, \tag{11}$$

where, for ease of notation, the dependency of time has been dropped. Mathematically this can be expressed by the following optimization problem:

$$\begin{aligned} \alpha^*_{\mathcal{B}} = \max_{\Delta \boldsymbol{p}} \ & \boldsymbol{n}^\top \boldsymbol{\Pi} \Delta \boldsymbol{p} \\ \text{s.t.} \ & -\Delta \boldsymbol{p}_{\max} \leq \Delta \boldsymbol{p} \leq \Delta \boldsymbol{p}_{\max}, \end{aligned} \tag{12}$$

where (7) has been used. Analyzing the cost function, it is possible to rewrite it as a sum of scalar elements:

$$\alpha_{\mathcal{B}} = \sum_{j=1}^{n_{\boldsymbol{p}}} \sum_{i=1}^{n_{\boldsymbol{x}}} n_i \Pi_{ij} \Delta p_j = \sum_{j=1}^{n_{\boldsymbol{p}}} a_j \Delta p_j, \tag{13}$$

where a_j is the j-th element of the vector $\boldsymbol{a} = \boldsymbol{n}^\top \boldsymbol{\Pi}$. By denoting as $\boldsymbol{a}^+$ and $\boldsymbol{a}^-$ the positive and negative elements of $\boldsymbol{a}$, it is possible to split (13) as:

$$\alpha_{\mathcal{B}} = \sum_{i \in \mathcal{A}^+} a_i^+ \Delta p_i + \sum_{j \in \mathcal{A}^-} a_j^- \Delta p_j, \tag{14}$$

where $\mathcal{A}^+$ and $\mathcal{A}^-$ are the set of the indexes of positive and negative elements of $\boldsymbol{a}$, respectively.

It is evident that the first sum is a monotonically increasing function if $\Delta p_i > 0$, while the second one is a monotonically increasing function if $\Delta p_j < 0$. Thus, it is possible to find the optimal solution, i.e. the maximum, $\alpha^*_{\mathcal{B}}$, as:

$$\alpha^*_{\mathcal{B}} = \sum_{i \in \mathcal{A}^+} a_i^+ \Delta p_{\max,i} + \sum_{j \in \mathcal{A}^-} a_j^- \left(-\Delta p_{\max,j}\right). \tag{15}$$

Indeed, the maximum contribution of each Δp_i and Δp_j is obtained at the boundary of the constraint, i.e. at one of the vertices of the hyperbox. Lastly, it is possible to combine the two terms in (15):

$$\begin{aligned} \alpha^*_{\mathcal{B}} &= \sum_{j=1}^{n_{\boldsymbol{p}}} |a_j| \Delta \boldsymbol{p}_{\max,j} = \sum_{j=1}^{n_{\boldsymbol{p}}} |\sum_{i=1}^{n_{\boldsymbol{x}}} n_i \Pi_{ij}| \Delta p_{\max,j} \\ &= \| \boldsymbol{n}^\top \boldsymbol{\Pi} \boldsymbol{W}^{\frac{1}{2}} \|_1, \end{aligned} \tag{16}$$

where the definition of 1-norm has been used. □

Despite its simplicity, the presence of the absolute value makes equation (8) non-differentiable, thus not suitable for gradient-based trajectory optimization algorithms (but it can be useful for gradient-free methods such as, e.g., MPPI). It is worth underlining that, for the hyperbox case, one may avoid directly computing the tube radius and may exploit equation (13) to build a set of $2^{n_{\boldsymbol{p}}}$ constraints. However, this exponentially increases the number of constraints, with a consequent increase in computational complexity.

B. Ellipsoid

One way to obtain a differentiable expression for α is to exploit ellipsoids. This method, firstly proposed in [11], is here revisited and extended with a formal proof, which was lacking in in [11].

The parameter space ellipsoid centered at $\boldsymbol{p}_n$ and inscribed[3] in (6) is defined as:

$$\boldsymbol{\mathcal{E}}_{\boldsymbol{p}} = \left\{ \Delta \boldsymbol{p} \in \mathbb{R}^{n_{\boldsymbol{p}}} \ : \ \Delta \boldsymbol{p}^\top \boldsymbol{W}^{-1} \Delta \boldsymbol{p} \leq 1 \right\} \simeq \boldsymbol{\mathcal{P}}. \tag{17}$$

Exploiting the sensitivity matrix $\boldsymbol{\Pi}$, it is possible to map (17) into the corresponding ellipsoid in the state space. For convenience, a proper linear scaling is first applied:

$$\boldsymbol{\sigma}_{\boldsymbol{p}} = \boldsymbol{W}^{-\frac{1}{2}} \Delta \boldsymbol{p} \tag{18}$$

$$\boldsymbol{P} = \boldsymbol{\Pi} \boldsymbol{W}^{\frac{1}{2}}, \tag{19}$$

such that the parameter ellipsoid defined in (17) becomes a sphere in the scaled parameter space:

$$\boldsymbol{\mathcal{S}}_{\boldsymbol{p}} = \left\{ \boldsymbol{\sigma}_{\boldsymbol{p}} \in \mathbb{R}^{n_{\boldsymbol{p}}} \ : \ \boldsymbol{\sigma}_{\boldsymbol{p}}^\top \boldsymbol{\sigma}_{\boldsymbol{p}} \leq 1 \right\}. \tag{20}$$

We underline that the linearization in (7) holds also for the scaled space and scaled sensitivity. Then, combining (18)-(19) leads to:

$$\Delta \boldsymbol{x} \simeq \boldsymbol{\Pi} \Delta \boldsymbol{p} = \boldsymbol{P} \boldsymbol{\sigma}_{\boldsymbol{p}}, \tag{21}$$

which can be pseudoinverted to compute:

$$\boldsymbol{\sigma}_{\boldsymbol{p}} = \boldsymbol{P}^\dagger \Delta \boldsymbol{x} + \boldsymbol{N}_{\boldsymbol{P}} \boldsymbol{w}, \tag{22}$$

where $\boldsymbol{N}_{\boldsymbol{P}} \in \mathbb{R}^{n_{\boldsymbol{p}} \times n_{\boldsymbol{p}}}$ is the nullspace projector of the scaled sensitivity matrix $\boldsymbol{P}$, and $\boldsymbol{w} \in \mathbb{R}^{n_{\boldsymbol{p}}}$ is a general solution projected in the nullspace.

By plugging (22) in (20), one can obtain the corresponding ellipsoid in the state space. Two possible cases arise depending on the dimensions of matrix $\boldsymbol{P}$. When $\boldsymbol{P}$ is tall or square, i.e. $n_{\boldsymbol{p}} \leq n_{\boldsymbol{x}}$, $\boldsymbol{N}_{\boldsymbol{P}} = \boldsymbol{0}$, leading to a unique solution. However, if $\boldsymbol{P}$ is fat, i.e. $n_{\boldsymbol{p}} > n_{\boldsymbol{x}}$, $\boldsymbol{N}_{\boldsymbol{P}} \neq \boldsymbol{0}$. Therefore, the parameter space ellipsoid can be mapped into an infinite number of state space ellipsoids, depending on $\boldsymbol{w}$.

Proposition 2. *The maximum state space ellipsoid inscribed in the hyperbox is defined as:*

$$\boldsymbol{\mathcal{E}}_{\boldsymbol{x}} = \left\{ \Delta \boldsymbol{x} \in \mathbb{R}^{n_{\boldsymbol{x}}} \ : \ \Delta \boldsymbol{x}^\top \boldsymbol{K}_{\boldsymbol{\Pi}}^\dagger \Delta \boldsymbol{x} \leq 1 \right\}, \tag{23}$$

where $\boldsymbol{K}_{\boldsymbol{\Pi}} = \boldsymbol{\Pi} \boldsymbol{W} \boldsymbol{\Pi}^\top$ represents the kernel matrix.

Proof. By plugging (22) in (20), one can obtain all the corresponding ellipsoids in the state space

$$\left(\boldsymbol{P}^\dagger \Delta \boldsymbol{x} + \boldsymbol{N}_{\boldsymbol{P}} \boldsymbol{w} \right)^\top \left(\boldsymbol{P}^\dagger \Delta \boldsymbol{x} + \boldsymbol{N}_{\boldsymbol{P}} \boldsymbol{w} \right) \leq 1. \tag{24}$$

Remembering that $\boldsymbol{P}^{\dagger^\top} \boldsymbol{N}_{\boldsymbol{P}} = \boldsymbol{0}$, since $\boldsymbol{P}^{\dagger^\top}$ maps any vector to the column space of $\boldsymbol{P}$ while $\boldsymbol{N}_{\boldsymbol{P}}$ annihilates all components in the column space, equation (24) becomes:

$$\Delta \boldsymbol{x}^\top \left(\boldsymbol{P} \boldsymbol{P}^\top \right)^\dagger \Delta \boldsymbol{x} + \boldsymbol{w}^\top \boldsymbol{N}_{\boldsymbol{P}}^\top \boldsymbol{N}_{\boldsymbol{P}} \boldsymbol{w} \leq 1. \tag{25}$$

[3]Although this leads to a more conservative result, the equations can also be directly applied to the circumscribed ellipsoid.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Since $\boldsymbol{w}^\top \boldsymbol{N}_{\boldsymbol{P}}^\top \boldsymbol{N}_{\boldsymbol{P}} \boldsymbol{w} \geq 0 \quad \forall \boldsymbol{w} \in \mathbb{R}^{n_{\boldsymbol{p}}}$, it is clear that by choosing any $\boldsymbol{w} \notin \ker(\boldsymbol{P})$, the volume of the ellipsoid would be reduced. Therefore, the largest ellipsoid is given by

$$\Delta \boldsymbol{x}^\top \boldsymbol{K}_{\boldsymbol{\Pi}}^\dagger \Delta \boldsymbol{x} \leq 1, \tag{26}$$

with $\boldsymbol{K}_{\boldsymbol{\Pi}} = \boldsymbol{P}\boldsymbol{P}^\top = \boldsymbol{\Pi}\boldsymbol{W}\boldsymbol{\Pi}^\top$. □

Regarding the tube radius, as detailed in [17], the maximum deviation of the ellipsoid with a kernel matrix $\boldsymbol{K}_{\boldsymbol{\Pi}}$ along a generic direction $\boldsymbol{n}$ can be computed as:

$$\alpha_{\boldsymbol{\mathcal{E}}} = \sqrt{\boldsymbol{n}^\top \boldsymbol{K}_{\boldsymbol{\Pi}} \boldsymbol{n}}, \tag{27}$$

which coincides with equation (9).

C. Superquadrics

As discussed previously, as the number of parameters increases, the ellipsoid covers a progressively smaller fraction of the hyperbox. To avoid this, in this paper we propose to exploit the *superquadrics*, a generalization of quadrics. In particular, among all the superquadrics a suitable approximation is given by the superellipsoids, defined by the following equation[4]:

$$\boldsymbol{\mathcal{Q}}_{\boldsymbol{p}} = \left\{ \Delta \boldsymbol{p} \in \mathbb{R}^{n_{\boldsymbol{p}}} \,:\, \sum_{i=1}^{n_{\boldsymbol{p}}} \left(\frac{\Delta p_i}{\Delta p_{\max,i}} \right)^{2k} \leq 1 \right\}, \tag{28}$$

where $2k$ is the degree of the superellipsoid, with $k \geq 1$. In particular, as k increases, the superellipsoid more closely approximates the shape of the hyperbox, especially around its corners; in contrast, when $k = 1$, equation (28) reduces to (17), yielding a regular ellipsoid. For simplicity, it is possible to express (28) in a vector form as:

$$\boldsymbol{\mathcal{Q}}_{\boldsymbol{p}} = \left\{ \Delta \boldsymbol{p} \in \mathbb{R}^{n_{\boldsymbol{p}}} \,:\, \Delta \boldsymbol{p}^{\circ k^\top} \boldsymbol{W}^{-k} \Delta \boldsymbol{p}^{\circ k} \leq 1 \right\} \simeq \boldsymbol{\mathcal{P}}, \tag{29}$$

where, as previously detailed, the superscript $\circ$ denotes the element-wise power.

As in the ellipsoid case, we apply a linear scaling such that the superquadric becomes a $2k$-norm ball [18]

$$\boldsymbol{\mathcal{S}}_{\boldsymbol{p}}^k = \left\{ \boldsymbol{\sigma}_{\boldsymbol{p}} \in \mathbb{R}^{n_{\boldsymbol{p}}} \,:\, \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ k^\top} \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ k} \leq 1 \right\}, \tag{30}$$

which is then mapped into state space by exploiting the sensitivity matrix. According to (22), the general solution to (21) is $\boldsymbol{\sigma}_{\boldsymbol{p}} = \left(\boldsymbol{P}^\dagger \Delta \boldsymbol{x} + \boldsymbol{N}_{\boldsymbol{P}} \boldsymbol{w} \right)$, from which one can obtain the state space superquadric:

$$\boldsymbol{\mathcal{Q}}_{\boldsymbol{x}} = \left\{ \Delta \boldsymbol{x} \in \mathbb{R}^{n_{\boldsymbol{x}}} : \left(\boldsymbol{P}^\dagger \Delta \boldsymbol{x} + \boldsymbol{N}_{\boldsymbol{P}} \boldsymbol{w} \right)^{\circ k^\top} \left(\boldsymbol{P}^\dagger \Delta \boldsymbol{x} + \boldsymbol{N}_{\boldsymbol{P}} \boldsymbol{w} \right)^{\circ k} \leq 1 \right\}. \tag{31}$$

Due to the element-wise power, it is easy to demonstrate that $\boldsymbol{w} \in \ker(\boldsymbol{P})$ is not the solution that maximizes the volume of the superquadric. This makes it not possible to explicitly define the state space superquadric a priori, depending instead on the unknown $\boldsymbol{w}$, and use it directly to construct tubes. Nonetheless, having defined the tube radius as the maximum directional deviation (cfr. Sect.III), an explicit expression for $\boldsymbol{\mathcal{Q}}_{\boldsymbol{x}}$ is not needed, as the tube radius α can be computed as follows.

[4]An alternative definition can be: $\sum_{i=1}^{n_{\boldsymbol{p}}} \left| \frac{\Delta p_i}{\Delta p_{\max,i}} \right|^k \leq 1$

Proposition 3. *The maximum deviation induced by the superquadric along a given direction $\boldsymbol{n}$ is*

$$\alpha_{\boldsymbol{\mathcal{Q}}} = \sqrt[\frac{2k}{2k-1}]{\boldsymbol{n}^\top \boldsymbol{P} \left(\boldsymbol{P}^\top \boldsymbol{n} \right)^{\circ \frac{1}{2k-1}}}. \tag{32}$$

Furthermore, as expected, $\alpha_{\boldsymbol{\mathcal{Q}}}$ converges to (8) *as $k \to \infty$.*

Proof. The goal is to obtain the maximum $\alpha_{\boldsymbol{\mathcal{Q}}}$ such that

$$\boldsymbol{n}^\top \Delta \boldsymbol{x} = \boldsymbol{n}^\top \boldsymbol{P} \boldsymbol{\sigma}_{\boldsymbol{p}} = \alpha_{\boldsymbol{\mathcal{Q}}} \;\wedge\; \boldsymbol{\sigma}_{\boldsymbol{p}} \in \boldsymbol{\mathcal{S}}_{\boldsymbol{p}}^k. \tag{33}$$

Since the $2k$-norm ball is a convex set, it is possible to focus on the boundary $\partial \boldsymbol{\mathcal{S}}_{\boldsymbol{p}}^k$. Thus, the optimal solution can be found by solving the following optimization problem:

$$\begin{aligned} \alpha_{\boldsymbol{\mathcal{Q}}}^* = \max_{\boldsymbol{\sigma}_{\boldsymbol{p}}} \;& \boldsymbol{n}^\top \boldsymbol{P} \boldsymbol{\sigma}_{\boldsymbol{p}} \\ \text{s.t.} \;& \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ k^\top} \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ k} = 1. \end{aligned} \tag{34}$$

As only equality constraints are present, problem (34) can be solved exploiting the Lagrangian Multipliers method. Firstly, the Lagrangian is defined as

$$\mathcal{L}(\boldsymbol{\sigma}_{\boldsymbol{p}}, \lambda) = -\boldsymbol{n}^\top \boldsymbol{P} \boldsymbol{\sigma}_{\boldsymbol{p}} + \lambda \left(\boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ k^\top} \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ k} - 1 \right), \tag{35}$$

and its gradient is

$$\begin{bmatrix} \nabla_{\boldsymbol{\sigma}_{\boldsymbol{p}}} \mathcal{L} \\ \nabla_{\lambda} \mathcal{L} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{P}^\top \boldsymbol{n} + 2k\lambda \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ 2k-1} \\ \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ k^\top} \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ k} - 1 \end{bmatrix}. \tag{36}$$

At this point, we compute the optimal $\alpha^* = \boldsymbol{n}^\top \Delta \boldsymbol{x}^*$ by imposing $\nabla \mathcal{L} = \boldsymbol{0}$, where for ease of notation $\boldsymbol{\sigma}_{\boldsymbol{p}}^* = \boldsymbol{\sigma}_{\boldsymbol{p}}$. Therefore:

$$\begin{cases} 2k\lambda \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ 2k-1} = \boldsymbol{P}^\top \boldsymbol{n}, & (37) \\ \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ k^\top} \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ k} = 1. & (38) \end{cases}$$

Left-multiplying (37) for $\boldsymbol{\sigma}_{\boldsymbol{p}}^\top$, and noting that $\boldsymbol{\sigma}_{\boldsymbol{p}}^\top \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ 2k-1} = \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ k^\top} \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ k}$, by substituting (38) one obtains

$$\lambda^* = \frac{\boldsymbol{\sigma}_{\boldsymbol{p}}^\top \boldsymbol{P}^\top \boldsymbol{n}}{2k} = \frac{\boldsymbol{n}^\top \boldsymbol{P} \boldsymbol{\sigma}_{\boldsymbol{p}}}{2k} = \frac{\alpha_{\boldsymbol{\mathcal{Q}}}^*}{2k}, \tag{39}$$

which plugged back in (37) yields

$$\alpha_{\boldsymbol{\mathcal{Q}}}^* \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ 2k-1} = \boldsymbol{P}^\top \boldsymbol{n}. \tag{40}$$

By transposing and applying the $(2k-1)$-th root to equation (40), one can obtain:

$$\alpha_{\boldsymbol{\mathcal{Q}}}^{*\,\frac{1}{2k-1}} \boldsymbol{\sigma}_{\boldsymbol{p}}^\top = \left(\boldsymbol{n}^\top \boldsymbol{P} \right)^{\circ \frac{1}{2k-1}}. \tag{41}$$

Finally, multiplying (40) by (41) results in

$$\alpha_{\boldsymbol{\mathcal{Q}}}^{*\,\frac{2k}{2k-1}} \boldsymbol{\sigma}_{\boldsymbol{p}}^\top \boldsymbol{\sigma}_{\boldsymbol{p}}^{\circ 2k-1} = \left(\boldsymbol{n}^\top \boldsymbol{P} \right)^{\circ \frac{1}{2k-1}} \boldsymbol{P}^\top \boldsymbol{n}, \tag{42}$$

which, recalling (38) and applying the $2k/(2k-1)$-th root, leads to the optimal solution $\alpha_{\boldsymbol{\mathcal{Q}}}^*$:

$$\begin{aligned} \alpha_{\boldsymbol{\mathcal{Q}}}^* &= \sqrt[\frac{2k}{2k-1}]{\left(\boldsymbol{n}^\top \boldsymbol{P} \right)^{\circ \frac{1}{2k-1}} \boldsymbol{P}^\top \boldsymbol{n}} \\ &= \sqrt[\frac{2k}{2k-1}]{\boldsymbol{n}^\top \boldsymbol{P} \left(\boldsymbol{P}^\top \boldsymbol{n} \right)^{\circ \frac{1}{2k-1}}}. \end{aligned} \tag{43}$$

It is worth highlighting that equation (43) returns exactly the ellipsoid solution in (9) when choosing $k = 1$, while it converges to the one of the hyperbox (8) when $k \to \infty$. Indeed:

$$\lim_{k \to \infty} \sqrt[\frac{2k}{2k-1}]{\boldsymbol{n}^\top \boldsymbol{P} \left(\boldsymbol{P}^\top \boldsymbol{n} \right)^{\circ \frac{1}{2k-1}}} = \boldsymbol{n}^\top \boldsymbol{P} \boldsymbol{S} = \| \boldsymbol{n}^\top \boldsymbol{\Pi} \boldsymbol{W}^{\frac{1}{2}} \|_1, \tag{44}$$

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

where $\boldsymbol{S} = \{\text{sign}\left(\boldsymbol{P}^\top \boldsymbol{n}\right)_i\} \in \mathbb{R}^{n_p}$. □

Therefore, by increasing the order k it is possible to obtain an expression that is still differentiable while reducing portion of unoccupied volume of the original hyperbox. Crucially, by computing the volume ratio of the superquadric and hyperbox [18], it is possible to choose k to meet a desired box covering ratio $r \in (0, 1)$ via

$$\frac{V_{\mathcal{Q}}}{V_{\mathcal{B}}} = \frac{\Gamma\left(1+\frac{1}{2k}\right)^{n_p}}{\Gamma\left(1+\frac{n_p}{2k}\right)} \geq r. \tag{45}$$

IV. EVALUATION

The proposed methods have been validated in two different scenarios via extensive numerical simulations that reproduce the effects of different parameter variations. The first case study involves a mass linearly moving freely in 3D space, while the second case study considers the dynamic behavior of a UR10e manipulator. In both cases, the sensitivity has been exploited to build an optimization problem computing a trajectory for point-to-point motion in configuration space. The optimization aims at minimizing the state sensitivity at the final time for improving accuracy and it leverages the input tubes to ensure that the actual (perturbed) inputs remain within the actuator limits, avoiding unwanted saturations which would result in a deviation from the planned motion. The optimization problem is formulated as:

$$\begin{aligned} \min_{\boldsymbol{a}} \quad & \|\boldsymbol{\Pi}\left(\boldsymbol{a}, t_f\right) \boldsymbol{W} \boldsymbol{\Pi}\left(\boldsymbol{a}, t_f\right)^\top\|_\gamma \\ \text{s.t.} \quad & \boldsymbol{A}_{\text{eq}} \boldsymbol{a} = \boldsymbol{b}_{\text{eq}}, \\ & \boldsymbol{u}_{\min} + \boldsymbol{\alpha}_{\boldsymbol{\Theta}}(\boldsymbol{a}) \leq \boldsymbol{u}(\boldsymbol{a}) \leq \boldsymbol{u}_{\max} - \boldsymbol{\alpha}_{\boldsymbol{\Theta}}(\boldsymbol{a}), \end{aligned} \tag{46}$$

where $\|\cdot\|_\gamma$ represents a generic matrix norm, e.g., 2-norm or Frobenius norm. $\boldsymbol{a} \in \mathbb{R}^{n_{\boldsymbol{a}}}$ are the coefficients that parametrize the trajectory, i.e. the coefficients of a polynomial parametrization, while $\boldsymbol{A}_{\text{eq}} \in \mathbb{R}^{n_{\text{eq}} \times n_{\boldsymbol{a}}}$ and $\boldsymbol{b}_{\text{eq}} \in \mathbb{R}^{n_{\boldsymbol{a}}}$ are the matrix and the vector encoding the boundary conditions, respectively. Lastly, $\boldsymbol{u}_{\min}$ and $\boldsymbol{u}_{\max}$ are the upper and lower input bounds, and $\boldsymbol{\alpha}_{\boldsymbol{\Theta}}$ is the stack of tube radii computed with the input sensitivity matrix $\boldsymbol{\Theta}$ in the direction of the i-th unit vector of the canonical basis $\boldsymbol{e}_i$, $\forall i \in [1, n_{\boldsymbol{u}}]$.

All the simulations have been performed in Matlab and can be found at the following Github repository[5].

A. 3D Mass

The goal of this simulation is to present a comparison between the two methods proposed in this paper, i.e. the tube computed with the hyperbox and the superquadric, showing that having a non-differentiable constraint may lead to failure of the optimization process. The optimization has been obtained exploiting CasADi and Ipopt solver.

Defining $\boldsymbol{z} \in \mathbb{R}^3$ and $\dot{\boldsymbol{z}} \in \mathbb{R}^3$ as the cartesian position and velocity of the mass respectively, and taking $\boldsymbol{x} = [\boldsymbol{z} \quad \dot{\boldsymbol{z}}]^\top$ as the system state, the 3D mass can be modeled as

$$\boldsymbol{f} = \begin{bmatrix} \dot{\boldsymbol{z}} \\ m^{-1}\left(\boldsymbol{u} - b\dot{\boldsymbol{z}}\right) \end{bmatrix} \tag{47}$$

[5]https://github.com/apupa/sensitivity_tubes.git

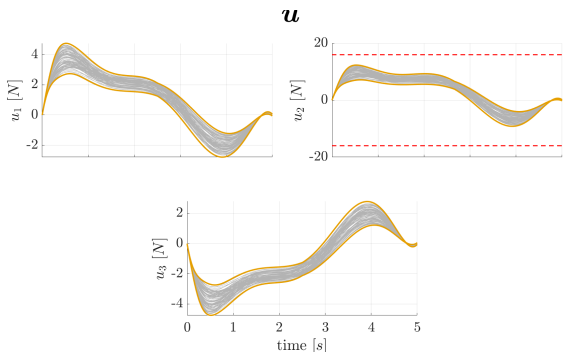

Fig. 3: Evolution of the input for the 3D Mass with the superquadric. The yellow lines represent the sensitivity tube, while the red lines illustrates the upper and lower bound of the input.

with $m \in \mathbb{R}$ and $b \in \mathbb{R}$ being the mass and the damping of the system, respectively, which are taken as the *uncertain parameters*.

To control the system, a PD + Friction compensation controller is employed:

$$\boldsymbol{u} = m_n\left(\ddot{\boldsymbol{z}}_{des} + \boldsymbol{K}_p\tilde{\boldsymbol{z}} + \boldsymbol{K}_d\dot{\tilde{\boldsymbol{z}}}\right) + b_n\dot{\boldsymbol{z}}. \tag{48}$$

Here, $m_n \in \mathbb{R}$ and $b_n \in \mathbb{R}$ are the *nominal* mass and damping, and $\boldsymbol{K}_p \in \mathbb{R}^{3\times3}$ and $\boldsymbol{K}_d \in \mathbb{R}^{3\times3}$ are the positive definite diagonal proportional and derivative gain matrices, respectively. The terms $\tilde{\boldsymbol{z}} = \boldsymbol{z}_{des} - \boldsymbol{z}$ and $\dot{\tilde{\boldsymbol{z}}} = \dot{\boldsymbol{z}}_{des} - \dot{\boldsymbol{z}}$ represent the error and its derivative, with $\boldsymbol{z}_{des}, \dot{\boldsymbol{z}}_{des}, \ddot{\boldsymbol{z}}_{des} \in \mathbb{R}^3$ being the desired position, velocity and acceleration of the trajectory.

The mass has to move from an initial configuration $\boldsymbol{z}_0 = [0.0, 0.0, 0.0]^\top$ to a desired final configuration $\boldsymbol{z}_f = [1.0, 3.0, -1.0]^\top$ in $t_f = 5\ s$, while both the initial and final velocity and acceleration are zero, minimizing the square of the Frobenius norm of the sensitivity at the end, i.e. $\gamma = F$. The nominal parameters are equal to $\boldsymbol{p}_n = [m_n, b_n]^\top = [10.0, 5.0]^\top$, while the real ones may vary within a range of 25% around their nominal values. The input limits are assumed equal in all the directions and symmetric, i.e. $u_{\max,i} = u_{\min,i} = 16.0\ N$. The optimization problem in (46) is solved computing α with the two different approaches proposed, i.e. the hyperbox defined in (8) and the superquadric defined in (10) with order $k = 50$.

Using the hyperbox constraint, the optimization problem is not capable of converging after $10,000$ iterations. However, it still finds a feasible suboptimal solution with a cost function value of $\|\boldsymbol{\Pi}(t_f)\boldsymbol{W}\Pi(t_f)^\top\|_F = 4.9\text{e}{-5}$. With the superquadric constraint, instead, the optimization problem converges in only 31 iterations, obtaining a better solution of $\|\boldsymbol{\Pi}(t_f)\boldsymbol{W}\Pi(t_f)^\top\|_F = 3.5\text{e}{-8}$. It is worth underlining that the solution obtained with the superquadric produces a lower cost while still being feasible for the hyperbox constraint. This means that, for this case, theoretically the optimization problem could converge to the same solution.

To further validate the sensitivity tubes, 100 simulations were performed with randomly varying parameters lying within the hyperbox. These simulations demonstrate that the tubes are capable of encapsulating all the perturbed state/input evolutions. In particular, Fig. 3 illustrates the input evolution for the superquadric case (the hyperbox is similar), showing

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

that the input never violates the constraint.

B. Dynamic Manipulator

The second simulation involves the dynamic model of a UR10e, a 6 DoF collaborative manipulator. The goal of this simulation is twofold. From one hand, it shows that the sensitivity approach can be applied also to more complex dynamics the full UR10e dynamic model. On the other hand, the simulation shows that the sensitivity tube computed through the superquadric is capable of encapsulating almost all the system evolution, outperforming the ellipsoid approximation exploited in the previous works. The optimization has been performed with fmincon using the interior-point algorithm. Let $\boldsymbol{x} = [\boldsymbol{q} \quad \dot{\boldsymbol{q}}]^\top$ be the state of the system, with $\boldsymbol{q}$ being the joint configuration. The robot is modeled exploiting the Euler-Lagrange formulation:

$$\boldsymbol{f} = \begin{bmatrix} \dot{\boldsymbol{q}} \\ \ddot{\boldsymbol{q}} \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{M}(\boldsymbol{q},\boldsymbol{p})^{-1}(\boldsymbol{\tau} - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{p})\dot{\boldsymbol{q}} - \boldsymbol{g}(\boldsymbol{q},\boldsymbol{p})) \end{bmatrix}. \tag{49}$$

where $\boldsymbol{M}(\boldsymbol{q},\boldsymbol{p}) \in \mathbb{R}^{6\times6}$ represents the inertia matrix at configuration $\boldsymbol{q}$, $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{p}) \in \mathbb{R}^{6\times6}$ is the Coriolis matrix, and $\boldsymbol{g}(\boldsymbol{q},\boldsymbol{p}) \in \mathbb{R}^6$ is the gravity vector. Lastly, $\boldsymbol{\tau} \in \mathbb{R}^6$ are the torque inputs. The set of parameters that are considered uncertain are the mass of each link. Since the robot manipulator has to follow a desired trajectory in the joint space, the following control law is employed:

$$\begin{aligned} \boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q},\boldsymbol{p}_n)\left(\ddot{\boldsymbol{q}}_{des} + \boldsymbol{K}_p\tilde{\boldsymbol{q}} + \boldsymbol{K}_d\dot{\tilde{\boldsymbol{q}}}\right) + \\ + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}},\boldsymbol{p}_n)\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q},\boldsymbol{p}_n), \end{aligned} \tag{50}$$

where the dependency of time has been dropped for ease of notation. $\ddot{\boldsymbol{q}}_{des} \in \mathbb{R}^6$ is the feedforward acceleration, while $\tilde{\boldsymbol{q}} = \boldsymbol{q}_{des} - \boldsymbol{q}$ and $\dot{\tilde{\boldsymbol{q}}} = \dot{\boldsymbol{q}}_{des} - \dot{\boldsymbol{q}}$ are the position and velocity error, respectively. Lastly, $\boldsymbol{K}_p \in \mathbb{R}^{6\times6}$ and the $\boldsymbol{K}_d \in \mathbb{R}^{6\times6}$ are the respective gains.

The robot is tasked to move from an initial configuration $\boldsymbol{q}_0 = \left[0.0, -\frac{3}{5}\pi, \frac{3}{5}\pi, -\frac{\pi}{2}, -\frac{\pi}{2}, 0.0\right]^\top$ to a desired final configuration $\boldsymbol{q}_f = \left[\frac{\pi}{2}, -\frac{3}{5}\pi, \frac{3}{5}\pi, -\frac{\pi}{2}, -\frac{\pi}{2}, 0.0\right]^\top$ in $t_f = 1$ s, while both the initial and final velocity and acceleration are zero. The nominal mass of each link can be found on the official UR website, while it is assumed that the real ones may vary within a range of 20% of their nominal one. The input limit is assumed equal in all the directions and symmetric, i.e. $u_{\max,i} = -u_{\min,i} = 23\ Nm$.

First, it is essential to quantify how using a superquadric approximation of the set impacts the estimation of the tube radius. This is done by defining the ratio $r_\alpha = \alpha_{\mathcal{Q}}/\alpha_{\mathcal{B}}$ as a function of the superquadric order k. Unlike the volume ratio in (45), r_α varies with the chosen direction $\mathbf{n}$ and the scaled sensitivity matrix $\boldsymbol{P}$, making a closed-form expression difficult to obtain. Consequently, a statistical analysis on the manipulator's input tubes is performed: for each of the 50 randomly generated trajectories, r_α is evaluated across all input directions along the entire path. Figure 4 shows boxplots of r_α for various values of k (left axis), while the dotted line traces the corresponding volume ratio from (45) (right axis). It is worth underlining that the ellipsoidal approximation employed in [11], i.e. $k = 1$, underestimates the tube radius of

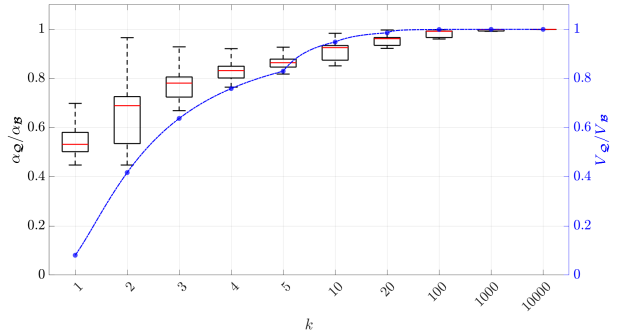

Fig. 4: Boxplots of the tube radius ratio (left) and the evolution of the volume ratio (right) for different values of k. The ellipsoidal approximation introduced in [11] clearly underestimates the maximum deviation and achieves low volume coverage.

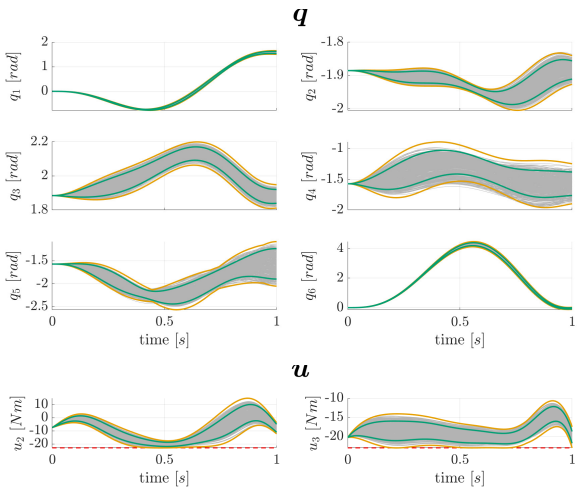

Fig. 5: Evolution of the manipulator joints (top) and inputs (bottom). The yellow lines represent the superquadric-based sensitivity tubes, while the green lines the ellipsoid-based ones. Lastly, the red lines illustrates the input upper and lower bound of the input. Only the inputs that saturate are shown for compactness, showing that the constraints are never violated.

approximately 50%, while the volume covered is lower than 10%.

Setting the order of the superquadric $k = 10$, the optimization problem minimizes the 2-norm in (46) in 163 iterations and the required time is around 3 minutes. Also in this case, to further validate the sensitivity tubes, 500 simulations were performed with randomly varying parameters lying within the hyperbox. These simulations demonstrate that the superquadric tubes are capable of encapsulating most of the perturbed state/input evolutions, shown in Fig. 5, while the ellipsoid tubes underestimate the variation.

To further validate the effectiveness of the superquadric approximation, the same optimization problem has been solved using ellipsoids, as in [11]. Fig. 6 illustrates that, in this case, the tubes fail to fully encapsulate the actual input evolutions. Consequently, some runs incur an unplanned input saturation with a negative impact on the joint evolution, showing significant deviations outside of the tubes.

Intuitively, this underestimation of the tube radius reduces the sensitivity tubes' ability to capture the system's closed-loop evolution. For this reason, another statistical campaign was carried out using 20 randomly generated trajectories.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

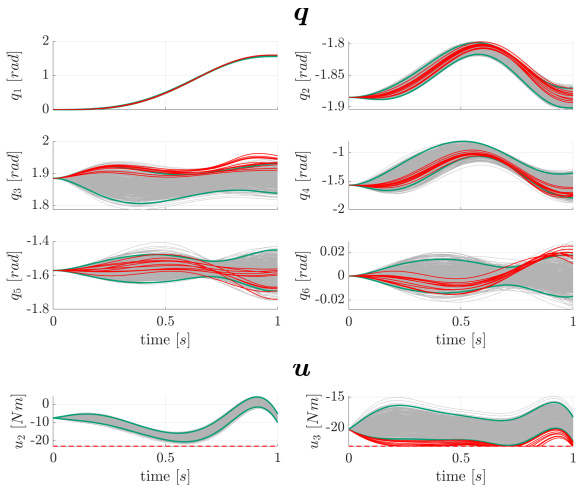

Fig. 6: Evolution of the manipulator joints (top) and inputs (bottom) with only the ellipsoid tubes. The solid red lines represent instances where the inputs have been saturated in order to respect the constraints. Only the inputs that saturate are shown for compactness.

Specifically, for each trajectory, 200 simulations were performed with the parameters sampled uniformly within the hyperbox, and it was analyzed wether the perturbed inputs remains within the tube computed using both ellipsoid and superquadric of order $k = 10$. To minimize numerical errors and remove linearization errors, the perturbed evolution was considered invalid if it never goes out of the hyperbox-based tube and if, at any time step, it exceeded the ellipsoid or superquadric tube by more than 2% of the maximum tube radius. The results show that with the ellipsoid 61% of the perturbed inputs go outside the tube, i.e. they are invalid, while with the superquadric only 4%.

V. CONCLUSIONS AND FUTURE WORK

In this work, two new ways for computing sensitivity tubes have been proposed: the first method exploits superquadrics for approximating a bounding hyperbox in parameter space with an arbitrary precision while retaining differentiability of the resulting tube radius, thus making it seamlessly usable in any gradient-based optimization algorithm. The second method, instead, exploits directly the definition of the bounding hyperbox for evaluating the maximum state deviation, which is proven to always lie at a vertex of the box, but loses full differentiability of the tube radius. For both methods, a formal proof is given for obtaining the corresponding tube radius, and the whole framework is validated in a simulation campaign on two case studies.

The proposed methods characterize the state/input deviations by exploiting the respective sensitivity matrices and assuming a linear behavior around the nominal parameters (7). Future work will focus on quantifying the linearization error and identifying an upper bound for its value. Consequently, the tube size could be increased to achieve robust guarantees. Moreover, the assumption of uniformly distributed parameter variations in a range could be replaced with other assumptions such as a Gaussian distribution centered at a nominal value. In this case, superquadrics could be designed to capture a desired probability of variations. Lastly, in real scenarios parameter variations are often correlated, e.g., a change in mass may also affect the position of the center of mass. The proposed method could be extended to account for such dependencies as well.

REFERENCES

[1] C. Zimmer, M. Meister, and D. Nguyen-Tuong, "Safe active learning for time-series modeling with gaussian processes," in *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[2] K. Astrom, "Adaptive feedback control," *Proceedings of the IEEE*, vol. 75, no. 2, pp. 185–217, 1987.

[3] K. Zhou and J. C. Doyle, *Essentials of robust control*. Prentice hall Upper Saddle River, NJ, 1998, vol. 104.

[4] Y. Shtessel, C. Edwards, L. Fridman, A. Levant *et al.*, *Sliding mode control and observation*. Springer, 2014, vol. 10.

[5] R. Lozano, B. Brogliato, O. Egeland, and B. Maschke, *Passivity-Based Control*. London: Springer London, 2000, pp. 227–277.

[6] A. van der Schaft, *L2-Gain and Passivity Techniques in Nonlinear Control*, 3rd ed. Springer Publishing Company, Incorporated, 2016.

[7] P. Robuffo Giordano, Q. Delamare, and A. Franchi, "Trajectory generation for minimum closed-loop state sensitivity," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 286–293.

[8] A. Srour, S. Marcellini, T. Belvedere, M. Cognetti, A. Franchi, and P. Robuffo Giordano, "Experimental validation of sensitivity-aware trajectory planning for a quadrotor uav under parametric uncertainty," in *2024 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, Jun. 2024, p. 572–578.

[9] A. Pupa, P. Robuffo Giordano, and C. Secchi, "Optimal energy tank initialization for minimum sensitivity to model uncertainties," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 8192–8199.

[10] S. Wasiela, P. Robuffo Giordano, J. Cortés, and T. Simeon, "A sensitivity-aware motion planner (samp) to generate intrinsically-robust trajectories," in *2023 IEEE International Conference on Robotics and Automation*, 2023.

[11] A. Afifi, T. Belvedere, A. Pupa, P. Robuffo Giordano, and A. Franchi, "Safe and robust planning for uncertain robots: A closed-loop state sensitivity approach," *IEEE Robotics and Automation Letters*, vol. 9, no. 11, p. 9962–9969, Nov. 2024.

[12] T. Belvedere, M. Cognetti, G. Oriolo, and P. Robuffo Giordano, "Sensitivity-aware model predictive control for robots with parametric uncertainty," *IEEE Transactions on Robotics*, vol. 41, pp. 3039–3058, 2025.

[13] A. H. Barr, "Superquadrics and angle-preserving transformations," *IEEE Computer graphics and Applications*, vol. 1, no. 01, pp. 11–23, 1981.

[14] G. Vezzani, U. Pattacini, and L. Natale, "A grasping approach based on superquadric models," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1579–1586.

[15] F. Messerer and M. Diehl, "An efficient algorithm for tube-based robust nonlinear optimal control with optimal linear feedback," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 6714–6721.

[16] R. Reams, "Hadamard inverses, square roots and products of almost semidefinite matrices," *Linear Algebra and its Applications*, vol. 288, pp. 35–43, 1999.

[17] D. H. Van Hessem, C. W. Scherer, and O. H. Bosgra, "Lmi-based closed-loop economic optimization of stochastic process operation under state and input constraints," in *2001 40th IEEE Conference on Decision and Control (CDC)*. IEEE, 2001, pp. 4228–4233.

[18] A. Slavík and D. Šulc, "Maximal volumes of n-dimensional balls in the p-norm," *Archiv der Mathematik*, vol. 114, no. 3, pp. 305–312, 2020.